COMMUNICATION AND NETWORKING RISER ECR FORM

Date: September 7, 2000

ECR# (assigned internally): #009

Name of Originator: **Brad Barmore**

Company Name: **Intel Corporation**

Originator Contact Information:
Phone: (503) 696-4393
e-mail: brad.barmore@intel.com

Title of the Change: **Update CNR Interface Version Compliance Algorithms**

Specification Title and Version: **CNR Specification, Version 1.0**

Reason for Change:
Currently the CNR Specification requires that BIOS compare verify that it is compatible with the CNR Compliance Register value. If a difference is found, then a message is to be displayed, indicating that functionality may be lost. As newer versions of the CNR Specification, or updates in the Specifications for the interfaces on the CNR are introduced, it becomes very difficult to determine (from a BIOS and system integrator standpoint) whether the BIOS understands what has changed. This ECN adds additional registers with in the CNR EEPROM, as well as changes the CNR BIOS algorithms to robustly handle future changes in the CNR and Interface Specification versions.

Description of Change:

This ECN affects several different sections of the CNR Specification. Each section that is changed is shown below, with the changes shown in red.

The basic idea of this change is to allow the BIOS algorithms to determine if it understands how to address and talk with the various interfaces and components attached to those interfaces. The method used to determine understanding is through the addition of registers in the configuration space for each interface. The new register is a Minimum Version Compliance Register. The value of this register is assigned based on the interface backward compatibility.

For example, if the AC ’97 Specification version is 2.1, but it is backward compatible with version 2.05, the AC ’97 Minimum Compliance would be programmed with a value of 2.05. This allows the BIOS to understand that the CNR carries components that are compliant with the V2.05 specification. When the BIOS identifies that it does not match the Minimum Compliance Register value, then a message is displayed, indicating the discrepancy, and the action to be taken.

The changes to the CNR specifications are shown below (in red):

		Odd Byte Address (i.e. 01h)								Even Address (i.e. 00h)								
Register Address	Register Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default Value
00h	EEPROM ID	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	9249h
02h	EEPROM Size	X	X	X	X	X	X	X	X	SZ7	SZ6	SZ5	SZ4	SZ3	SZ2	SZ1	SZ0	N/A
04h	CNR Compliance	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0110h
06h	AC '97 Compliance	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0210h
08h	Function ID	X	X	X	X	X	X	X	X	X	X	X	LAN	SMB	USB	MDM	AUD	00xxh
0Ah to 0Ch	Reserved Registers	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0000h
0Eh	Audio Pointer	AP15	AP14	AP13	AP12	AP11	AP10	AP9	AP8	AP7	AP6	AP5	AP4	AP3	AP2	AP1	AP0	0000h
10h	Modem Pointer	MP15	MP14	MP13	MP12	MP11	MP10	MP9	MP8	MP7	MP6	MP5	MP4	MP3	MP2	MP1	MP0	0000h
12h	USB Pointer	UP15	UP14	UP13	UP12	UP11	UP10	UP9	UP8	UP7	UP6	UP5	UP4	UP3	UP2	UP1	UP0	0000h
14h	SMBus Pointer	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	0000h
16h	LAN Pointer	LP15	LP14	LP13	LP12	LP11	LP10	LP9	LP8	LP7	LP6	LP5	LP4	LP3	LP2	LP1	LP0	0000h
18h to 2Ch	Reserved Pointers	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0000h
2Eh	Last Valid Address	LV15	LV14	LV13	LV12	LV11	LV10	LV9	LV8	LV7	LV6	LV5	LV4	LV3	LV2	LV1	LV0	N/A
30h	Checksum	CS15	CS14	CS13	CS12	CS11	CS10	CS9	CS8	CS7	CS6	CS5	CS4	CS3	CS2	CS1	CS0	N/A

Table 1 – CNR EEPROM Master Configuration Space Register Map

6.1.2.3 CNR Compliance Register (Word 04h)

The CNR compliance register provides a 16-bit value that identifies the version of the CNR specification that the CNR board is compliant with. The BIOS uses this value to verify that the motherboard supports the features and version of the PnP EEPROM present on the CNR board.

The value of this register can be easily created by placing the numbers to the left of the decimal point in the CNR specification version number in the upper byte of the CNR compliance register (address 05h). Then, place the numbers to the right of the decimal point in the lower byte of the CNR compliance register (address 04h). See the example in Table 2.

The CNR Compliance Register must contain the value 0110h to be compliant with this release of the CNR Specification.

CNR Specification Version	CNR Compliance Register Upper Byte (Address 05h)	CNR Compliance Register Lower Byte (Address 04h)
0.80	00h	80h
1.00	01h	00h
1.10	01h	10h
1.53	01h	53h

Table 2 – CNR Specification Version Mapping in PnP EEPROM

6.1.2.10 SMBus Pointer Register (Word 14h)

The SMBus pointer register is a 16-bit register containing a pointer to the first address of the PnP EEPROM containing SMBus function data for the SMBus solution on the CNR board. The BIOS uses this value, along with the next non-zero pointer value, to determine how many words to read from the PnP EEPROM for the SMBus function.

~~Note: In this version of the CNR Specification, the SMBus pointer register must always be programmed as 0000h.~~

6.1.2.4 AC '97 Compliance Register (Word 06h)

The AC '97 Compliance Register provides a 16-bit value that identifies the version of the AC '97 ~~specification~~ Specification, BIOS must be compliant with, in order to properly access and determine the functionality of the codecs on the CNR board. ~~the codecs on the CNR board are compliant with. The BIOS uses this value to verify that the motherboard devices can support the AC '97 codecs present on the CNR.~~

The value contained in this register is the oldest version of the AC '97 Specification that is interface and register set compatible with the current version of the AC '97 Specifcation. Table 3 provides examples of the values to be placed in the AC '97 Complaince Register, based on the currently released versions of the AC '97 Specification.

~~The value of this register can be easily created by placing the numbers to the left of the decimal point in the AC '97 specification version number in the upper byte of the AC '97 compliance register (address 07h). Then, place the numbers to the right of the decimal point in the lower byte of the AC '97 compliance register (address 06h). See the example in Table 3.~~

Current AC '97 Specification Version	Backward Compatable with AC '97 Specification Version	AC '97 Compliance Register Upper Byte (Address 07h)	AC '97 Compliance Register Lower Byte (Address 06h)
1.03	1.03	01h	03h
2.0	2.0	02h	00h
2.1	2.1	02h	10h
2.2	2.1	02h	10h

Table 3 – AC '97 Compliance Register Contents

6.1.5 USB Function PnP EEPROM Contents

This area of the PnP EEPROM is reserved for the USB function. The intent of this section is that it contains the information required to identify basic USB feature set support to the BIOS.

Note: If the USB pointer register is non-zero, then this section of the EEPROM must be implemented.

The starting address for the USB function data can be located anywhere within the PnP EEPROM memory space, as long as it starts at the address defined by the USB pointer register (word 12h). The USB function section uses a maximum of ~~one word~~ two words (~~two~~ four bytes).

The following table describes the contents of the USB function section of the PnP EEPROM.

		Odd Byte Address (i.e. 01h)								Even Address (i.e. 00h)								
Register Address	Register Name	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default Value
UP+000h	USB Option Register	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	HUB	WKE	SPD1	SPD0	000xh
UP+002h	USB Compliance	UC15	UC14	UC13	UC12	UC11	UC10	UC9	UC8	UC7	UC6	UC5	UC4	UC3	UC2	UC1	UC0	0000h

Table 4 – CNR EEPROM USB Section Register Map

6.1.5.1 USB Option Register (Word UP+000h)

The USB option register contains individual bits that describe the various functions supported by the USB function implemented on the CNR board. The following paragraphs describe each bit implemented in the USB option register. Note that bits 15 through 4 are reserved for future use and must be programmed as zero.

HUB - Bit 3 - When set (1), indicates that the USB function implemented on the CNR board is a USB hub device. When cleared (0), indicates that the USB function implemented on the CNR board is a function other than a USB hub device. The BIOS uses this bit, in conjunction with the other bits in the USB option register, to determine if the motherboard is capable of supporting the functions implemented on the CNR board.

WKE - Bit 2 - When set (1), indicates that the USB function on the CNR Board is capable of supporting a wake-up event. When cleared (0), the USB function on the CNR board is not capable of supporting a wake-up event. The BIOS uses this bit to determine if the motherboard can support a wake event from the USB function on the CNR board, from both motherboard feature support and power supply delivery standpoints.

SPD[1:0] - Bits 1 and 0 - These bits indicate the version of USB required to support the intended USB function and feature set on the CNR Board (whether ~~it be~~ a hub, device, function or port ~~direct routing of the USB signals from the CNR connector to a port on the CNR I/O bracket~~). For example, if the function on the CNR card required USB2.0 high speed mode to meet the advertised functionality, then the SPD bits would be set to 10b. The BIOS must verify that the USB signals routed to the CNR connector support the same USB ~~version~~ mode ~~that is~~ called out by the SPD[1:0] bits. The following table describes the SPD[1:0] bit assignments for the various modes ~~version~~ of USB1.x and USB2.0.

USB Version	SPD1	SPD0
USB1.x – Low Speed	0	0
USB1.x – Full Speed	0	1
USB2.0 – High Speed	1	0
Reserved	1	1

Table 5 – USB Version to SPD Bit values

6.1.5.2 USB Compliance Register (Word UP+002h)

The USB Compliance Register provides a 16-bit value that identifies the version of the USB Specification, BIOS must be compliant with, in order to properly access and determine the functionality of the USB devices or functions on the CNR board.

The value contained in this register is the oldest version of the USB Specification that is interface and register set compatible with the current version of the USB Specifcation. Table 26 provides examples of the values to be placed in the USB Complaince Register, based on the released version of the USB Specification, at the time of this specification's publication.

Current USB Specification Version	Backward Compatable with USB Specification Version	USB Compliance Register Upper Byte (Address UP+003h)	USB Compliance Register Lower Byte (Address UP+002h)
1.0	1.0	01h	00h
1.1	1.0	01h	00h
2.0	2.0	02h	00h

Table 26 – USB Compliance Register Contents

6.1.6 SMBus Section EEPROM Map

This area of the PnP EEPROM is reserved for the SMBus function. This section describes the information required to identify basic SMBus feature set support to the BIOS.

Note: If the SMBus pointer register is zero, then this section of the EEPROM must not be implemented.

The starting address for the SMBus function data can be located anywhere within the PnP EEPROM memory space, as long as it starts at the address defined by the SMBus pointer register (word 14h). The SMBus function section uses a maximum of ~~two~~ three words (~~four~~ six bytes).

The following table describes the contents of the SMBus function section of the PnP EEPROM.

		Odd Byte Address (i.e. 01h)								Even Address (i.e. 00h)								
Register Address	**Register Name**	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default Value**
SP+000h	**SMBus Compliance**	**SC15**	**SC14**	**SC13**	**SC12**	**SC11**	**SC10**	**SC9**	**SC8**	**SC7**	**SC6**	**SC5**	**SC4**	**SC3**	**SC2**	**SC1**	**SC0**	**0000h**
SP+002h to SP+004h	**SMBus Function Reserved**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0000h**

Table 6 – CNR EEPROM SMBus Section Register Map

6.1.6.1 SMBus Compliance Register (Word SP+000h)

The SMBus Compliance Register provides a 16-bit value that identifies the version of the SMBus Specification BIOS must be compliant with, in order to properly access and determine the functionality of the SMBus PnP EEPROM device on the CNR board.

The value contained in this register must indicate System Management Bus, Revision 1.1 (dated December 11, 1998). Table 28 provides examples of the values to be placed in the SMBus Complaince Register, based off of currently released versions of the SMBus Specification.

CNR does not currently support Revision 2.0 of the System Management Bus Specification due to the limited availability of SMBus 2.0 devices and controllers. When SMBus 2.0 devices and controllers are widely available, the CNR Specification will be revised, if necessary.

Current SMBus Specification Version	Backward Compatable with SMBus Specification Version	SMBus Compliance Register Upper Byte (Address SP+003h)	SMBus Compliance Register Lower Byte (Address SP+002h)
1.0	1.0	01h	00h
1.1	1.0	01h	00h
2.0	2.0	02h	00h

Table 28 – SMBus Compliance Register Contents

6.1.6.2 SMBus Function Reserved (Words SP+~~000h~~002h to SP+~~002h~~004h)

These two reserved 16-bit words occupy the address space starting at the SMBus pointer register plus ~~000h~~ 002h, and continue up to the SMBus pointer register plus ~~002h~~ 004h. The SMBus function reserved registers are reserved for future feature support.

6.1.7 LAN Section EEPROM Map

This area of the PnP EEPROM is reserved for the LAN function. This section describes the information required to identify basic LAN feature set support to the BIOS.

Note: If the LAN pointer register is non-zero, then this section of the EEPROM must be implemented.

The starting address for the LAN function data can be located anywhere within the PnP EEPROM memory space, as long as it starts at the address defined by the LAN pointer register (word 16h). The LAN function section uses a maximum of ~~three~~ four words (~~six~~ eight bytes).

The following table describes the contents of the LAN function section of the PnP EEPROM.

		Odd Byte Address (i.e. 01h)								Even Address (i.e. 00h)								
Register Address	**Register Name**	**Bit 15**	**Bit 14**	**Bit 13**	**Bit 12**	**Bit 11**	**Bit 10**	**Bit 9**	**Bit 8**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Default Value**
LP+000h	**LAN Option Register**	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	**INTF**	**0000h**
LP+002h	**LAN CNR Vendor ID**	**LV15**	**LV14**	**LV13**	**LV12**	**LV11**	**LV10**	**LV9**	**LV8**	**LV7**	**LV6**	**LV5**	**LV4**	**LV3**	**LV2**	**LV1**	**LV0**	**0000h**
LP+004h	**LAN CNR Model ID**	**LM15**	**LM14**	**LM13**	**LM12**	**LM11**	**LM10**	**LM9**	**LM8**	**LM7**	**LM6**	**LM5**	**LM4**	**LM3**	**LM2**	**LM1**	**LM0**	**0000h**
LP+006h	**LAN Compliance**	**LC15**	**LC14**	**LC13**	**LC12**	**LC11**	**LC10**	**LC9**	**LC8**	**LC7**	**LC6**	**LC5**	**LC4**	**LC3**	**LC2**	**LC1**	**LC0**	**0000h**

Table 7 – CNR EEPROM LAN Section Register Map

6.1.7.1 LAN Option Register (Word LP+000h)

The LAN option register contains individual bits that describe the required support of the LAN function implemented on the CNR board. The following paragraphs describe each bit implemented in the LAN option register. Note that bits 15 through 1 are reserved for future use and must be programmed as zero. The physical address for this data begins at the address defined by the LAN pointer register.

INTF - Bit 0 - When set (1), indicates that the LAN function implemented on the CNR board requires the MII (17-pin) interface (as implemented on the Type B CNR connector). When cleared (0), indicates that the LAN function implemented on the CNR board requires the eight-pin LAN Interface (as implemented on the Type A CNR connector). The BIOS uses this bit to determine if the motherboard is capable of supporting the LAN interface required by the LAN device implemented on the CNR board.

6.1.7.2 LAN CNR Vendor ID (Word LP+002h)

This register contains a 16-bit value that indicates the subsystem vendor ID register of the LAN function. The actual value placed in this location should be the PCI SIG assigned vendor ID number for the company that is providing the completed CNR board and drivers. The physical address for this data begins at the address defined by the LAN Pointer Register plus 002h.

6.1.7.3 LAN CNR Model ID (Word LP+004h)

This register contains a 16-bit value that indicates the subsystem ID register of the LAN function. The actual value placed in this location should be a unique (to the manufacturer of the CNR board) model number. The physical address for this data begins at the address defined by the LAN pointer register plus 004h.

6.1.7.4 LAN Compliance Register (Word LP+006h)

The LAN Compliance Register provides a 16-bit value that identifies the LAN Interface version that BIOS must be compliant with, in order to properly access and determine the functionality of the LAN components on the CNR board.

The value contained in this register is dependent on the LAN Interface used by the LAN device on the CNR. In the case of the eight-bit LAN Interface, the value is related to the Intel I/O Controller Hub (ICH) chip, while in the case of the seventeen-bit LAN Interface the value is related to the IEEE802.3(u) Standard date. Tables 30 and 31 provide examples of the values to be placed in the LAN Compliance Register.

Intel I/O Controller Hub Part Number	***SMBus Compliance Register Upper Byte (Address LC+003h)***	***SMBus Compliance Register Lower Byte (Address LC+002h)***
82801AA	*00h*	*00h*
82801AB	*00h*	*00h*
82801BA	*02h*	*A0h*

Table 30 – LAN Compliance Register Contents for eight-pin LAN Interface

IEEE802.3u Specification Version Date	*LAN Compliance Register Upper Byte (Address LC+003h)*	*LAN Compliance Register Lower Byte (Address LC+002h)*
June 14, 1995	*00h*	*A4h*
September 28, 1998	*05h*	*56h*

Table 31 – LAN Compliance Register Contents for seventeen-pin LAN Interface (MII)

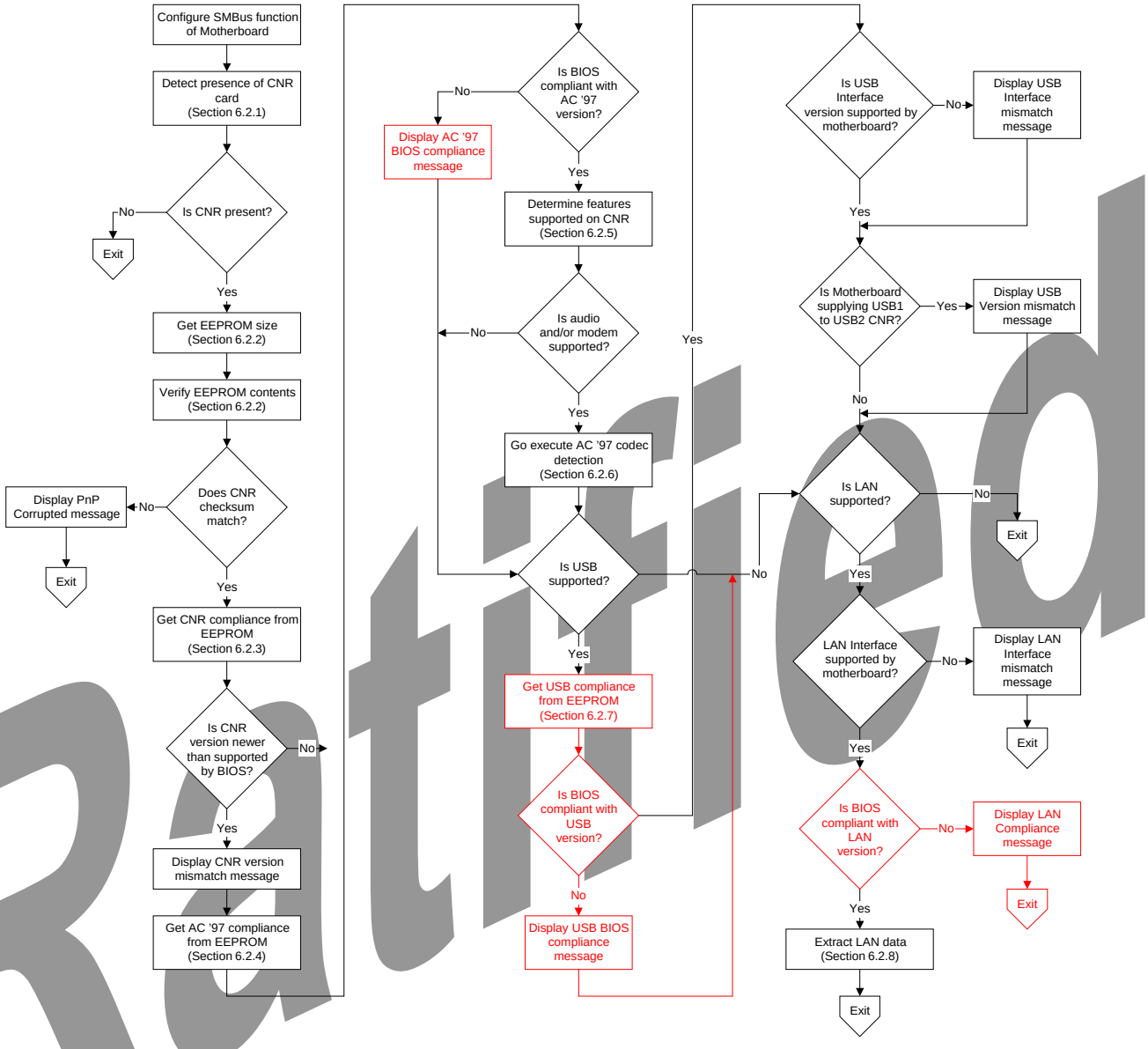

Figure 27 – BIOS Functional Flowchart for CNR Plug-and-Play Support

6.2.4 Verifying AC ’97 Version Compliance

After verifying compatibility with the PnP EEPROM version, the BIOS must then make sure that it understands the version of AC ’97 required by the codecs on the CNR. The BIOS performs this function by checking the AC '97 ~~compliance register~~ Compliance Register of the PnP EEPROM, as follows

The following steps assume that the previous steps have been completed.

1. Perform a direct read of addresses 06h and 07h. Concatenate the contents of these two registers together (with contents of address 07h as the most significant byte).
2. The BIOS **must** then verify that it is compliant with ~~compare~~ the resulting AC '97 Compliance value ~~to the version of the AC '97 Specification it is compliant with~~.
3. If BIOS is not compliant with the resulting number ~~is not the same as the BIOS AC '97 version number, then the AC '97 controller providing the AC-link to the CNR connector must be disabled. The~~ the error message "**BIOS not compliant with the AC '97 devices on CNR ~~and AC '97 Version do not match~~. AC '97 functionality of CNR disabled ~~ignored~~.**" must then be displayed indicating the version mismatch, and the audio and~~/or~~ modem functions of the AC '97 controller function(s) are being disabled.
4. If the resulting number is the same as the BIOS CNR version number then the BIOS can continue with the boot process.

   Note: If compatibility between the AC '97 codecs on the CNR and the controller on the motherboard cannot be established, the BIOS must not allow AC '97 codecs to be recognized, as this will cause driver installation and compatibility issues.

6.2.7 USB Compliance and Options

The USB interface, by definition, contains all of the necessary protocol to retrieve the Plug-and-Play information for any USB function attached to the USB signals on the CNR board. However, there may be instances where a USB device, on a CNR board, requires motherboard features that have not been implemented on the motherboard for the CNR connector. Some of these features include power supply current (for wake capabilities) or a USB 1.x compliant port routed to the CNR connector, when a USB 2.x compliant function is installed on the CNR.

In order to determine these type of feature differences, the BIOS **must** retrieve data from the USB option register, as follows.

1. Read, from the PnP EEPROM, byte addresses 12h and 13h. Concatenate the resulting two reads (with the contents of address 13h as the most significant byte). The resulting concatenation becomes the USB pointer.
2. The BIOS then continues to read the successive pointer locations until it encounters a non-zero pointer. When this non-zero pointer is encountered (or the last valid address register is reached, at word 2Eh), the BIOS subtracts the USB pointer from the non-zero pointer. The resulting answer is the number of bytes to be read from the PnP EEPROM for the USB function.

   Note: For this version of the CNR specification, this number of bytes must be two (one 16-bit word).

3. BIOS then reads the first two bytes of USB ~~option~~ data from the address pointed to by the USB pointer (from Step 1 of this section). These two bytes are concatenated (with the second byte read being the most significant byte), producing the USB Option Register contents. The value extracted for the USB option Register should be temporarily stored, until the Step 4 is completed.
4. BIOS then reads the next two bytes of the USB data (addresses UP+002h and UP+003h). These two bytes are concatenated together (with the second byte read being the most significant byte). The resulting concatenation indicates the minimum level of BIOS compliance required to be able to support the USB devices or functions on the CNR. If BIOS is not able to support the devices or functions on the CNR, the message "**BIOS not compliant with USB devices on CNR.**" must be displayed.
5. BIOS now uses the value previously extracted for the USB Option Register and with ~~With~~ the exception of the SPD[1:0] bits, the resulting concatenation can then be decoded and used as described in Section 0.
6. The BIOS may use the SPD[1:0] bits to determine if the motherboard supports the USB version required by the CNR. If the USB version required by the CNR is greater than supported by the motherboard, the BIOS may display the message "**USB version required by the CNR is not supported by motherboard. The CNR USB function will operate at a lower speed.**" If the USB version required by the CNR is lower than supported by the motherboard, then the BIOS may display a message indicating the discrepancy.
7. At this point BIOS has read the correct number of bytes (two) and can continue with the balance of the CNR BIOS code.

6.2.8 LAN Options

Extraction of the LAN Plug-and-Play information from the CNR PnP EEPROM is very straightforward. The BIOS first verifies that the motherboard supports the type of LAN interface required by the CNR board, then verifies that it compliant with the LAN Interface version, and finally~~, then simply~~ copies the information from the CNR PnP EEPROM to the appropriate locations within PCI configuration space for the LAN function, as described in the following steps.

1. Read, from the PnP EEPROM, byte addresses 16h and 17h. Concatenate the resulting two reads together (with the contents of address 17h as the most significant byte). The resulting concatenation becomes the LAN pointer.
2. The BIOS then continues to read the successive pointer locations until it encounters a non-zero pointer. When this non-zero pointer is encountered (or the last valid address is reached, word 2Eh), the BIOS subtracts the LAN pointer from the non-zero pointer. The resulting answer is the number of bytes to be read from the PnP EEPROM for the LAN function.

Note: For this version of the CNR specification, this number of bytes must be ~~six~~ eight (~~three~~ four 16-bit words).

3. BIOS then reads the first two bytes if the LAN function PnP data from the address pointed to by the LAN pointer (from step 1 of this section). These two bytes are concatenated (with the second byte being the most significant byte).
4. The least significant bit (INTF bit) of the resulting concatenation is then used to determine if the LAN interface required by the CNR board is the same as is provided by the motherboard (refer to Section 0 for information regarding the decoding of the INTF bit). If the LAN interfaces do not match, the BIOS must disable the MAC connected to the motherboard LAN interface (effectively removing the MAC from the PCI bus). The BIOS **must** also display the message "**CNR LAN Interface not compatible with Motherboard LAN Interface. LAN Function has been disabled.**" indicating the mismatch to the user.
5. BIOS then reads the next two bytes of LAN function PnP data. These two bytes are concatenated (with the second byte read being the most significant byte). The resulting concatenation is then stored for future use.
6. ~~The resulting concatenation can then be written to the subsystem vendor ID (SVID) register of the PCI configuration space for the LAN controller (if needed)~~
7. BIOS then sequentially reads the next two bytes of LAN function PnP data from the CNR EEPROM. These two bytes are concatenated (with the second byte read being the most significant byte). The resulting concatenation is then stored for future use.
8. BIOS then sequentially reads the next two bytes of LAN function PnP data from the CNR EEPROM. These two bytes are concatenated (with the second byte read being the most significant byte). BIOS uses this value to determine if it is compliant with the interface version required by the LAN devices on the CNR. If BIOS is not compliant the message **"BIOS not compliant with CNR LAN devices. LAN function disabled.**"
9. If BIOS is compliant with the LAN Interface, then the LAN PnP data extracted in Step 5 is written to the subsystem vendor ID (SVID) register of the PCI configuration space for the LAN controller (if needed)
10. BIOS then takes the LAN function PnP data extracted in Step 7 and writes it ~~The resulting concatenation is then written~~ to the subsystem ID (SID) register of the PCI configuration space for the LAN controller (if needed).
11. At this point, BIOS has read the correct number of bytes (~~six~~ eight), and the BIOS code can continue with the balance of the CNR BIOS code.